LNW-CP/M OPERATING SYSTEM

by R.M. Stiles

GENERAL DESCRIPTION

LNW-CP/M was designed to be compatible with application programs written for CP/M 2.2 (CP/M is a trade mark of Digital Research corp).

LNW-CP/M loads at D800H with BIOS at EE00H. This provides the user with a 61K system.

LNW-CP/M Disk system permits both single and double density, 5 and/or 8 inch disks, however, it is a double density environment and system disks have to be double density. A utility program called LNW Utility allows disk table changes to be set for a variety of disk formats. The LNW-CP/M System disk must be on one of the following:

5 1/4"	40-Track	Single Sided	Double Density
5 1/4"	40-Track	Double Sided	Double Density
5 1/4"	80-Track	Single Sided	Double Density
5 1/4"	80-Track	Double Sided	Double Density
8"	77-Track	Single Sided	Double Density
8"	77-Track	Double Sided	Double Density

The LNW Utility program allows selection of any of these formats, plus a variety of other popular formats. The utility program "SET" allows the knowledgeable user to manually change the disk parameters and sector skew tables for any standard formated disk system.

This text will briefly describe the standard CP/M programs, and fully describe the programs unique to LNW/CPM. For a full description of CP/M, see the CP/M operations manual.

BRINGING UP LNW/CPM:

Power up the LNW-II according to procedures outlined in the LNW user manual. WARNING -- Disk should not be in the drives during power up or power down. Insert your LNW/CPM disk in drive "A" and push the two reset keys. The disk size switch on the 5/8 doubler must be in the proper position (refer to LNW operation manual). The screen should clear and the sign-on message should appear while the system is loading. The screen will again clear and the prompt " A> " with blinking cursor will appear. The LNW/CPM operating system is now in operation. Type "DIR" and enter to display programs and files on the disk. See appendix (A) for a list of files supplied on your distribution diskette.

BACKING UP DISKETTES:

The first thing that you should do is to make a back-up of your Distribution Diskette. We suggest that you make two back-ups, storing your original and one back-up called your "Master Back-up" in a safe place and using your second back-up for normal

use. The following procedure will create your back-up:

1. Insert your original in drive "A" and boot system.

2. Insert a blank diskette to be used for a back-up into drive "B".

3. Type "FMT"and press enter key (future instructions will assume the proper use of the enter key).

4. You will be asked which drive. Respond with the drive name with the disk to be backed up.

5. You will now have a menu listing all formats available. Select the one for your system.

6. You now have a chance to review your selection, if the selection is correct, respond with "Y". If not, then "N" will return you to the beginning.

7. The system will now format your disk. It does each track in three steps: write, read, and verify.

8. If any of the three fail, an error message will appear. This usually means a bad diskette. You can try again or use another diskette.

9. At the end of the formatting, the system will read the system tracks from your master and then write them to your back-up.

10. At the completion message type "X". This will return you to the operating system prompt. You now have a back-up of the "system " but no files. Use the CP/M PIP command to copy files.

```
PIP B:=A:*.*[OV]
```

The open bracket is the (f1) key and the close bracket is the (f2) key. These appear on the screen as the (↑) for ([) and (→)for (]). This command will copy all files on drive "A" your original to drive "B" your back-up. The [OV] will be explained in the section under PIP.

On completion, you will have a back-up of your distribution diskette. Put it away and use your master for all other back-ups.

The LNW utility program should now be used to configure the disk for your specific system characteristics. This should then be saved to your operating disk to permit you to boot the system with the acquired characteristics rather than go though the process each time.

LNW-CP/M 2.2 system provides rapid access to programs through a comprehensive file management manager, the file sub system supports a named file structure, allowing dynamic allocation of file space as well as sequential and random file access. Using this file system, a large number of programs can be stored in both source and machine executable form.

Users of LNW CP/M 2.2 are physically separated by user numbers with facilities for file copy operations from one user area to another.

LNW CP/M 2.2 is logically divided into several distinct parts:

BIOS	-	Basic I/O system
BDOS	-	Basic disk operating system
CCP	-	Console command processor
TPA	-	Transient program area

The BIOS provides the primitive operations necessary to access the diskette drives and to interface with the console and the printer.

The BDOS provides disk management by controlling one or more disk drives containing independent file directories. The BDOS implements disk allocation strategies that provide full dynamic file construction while minimizing head movement across the disk during access.

The CCP provides a symbolic interface between the users console and the remainder of the CP/M system. The CCP reads the console and processes commands. The standard commands that are available in the CCP are listed in a later section.

The last segment of CP/M is the area called the Transient Program Area (TPA). The TPA holds programs that are loaded from the disk under command of the CCP.

Any or all of the CP/M component sub-systems can be "overlaid" by an executing program. That is, once a user's program is loaded into the TPA, the CCP, BDOS, and BIOS areas up to F700H can be used as the program's data area. The area from F700H to FFFFH are the memory mapped areas and, cannot be overlaid. A "bootstrap" loader is programmatically accessible whenever the BIOS portion (EE00H) is not overlaid; thus, the user program need only branch to the bootstrap loader at the end of execution and the complete CP/M monitor is reloaded from disk. If the BIOS is overlaid then the system has to be "cold started".

FUNCTIONAL DESCRIPTION:

The user interacts with LNW CPM through the CCP, which reads and interprets commands entered through the console. The CCP addresses one of the disks that are on-line (the system supports up to four drives). These disks are labeled A through D. The disk is "logged in" if the CCP is currently addressing the disk. To clearly indicate which disk is the currently logged-in disk, the CCP always prompts the operator with the disk name followed by the symbol" > " indicating that the CCP is ready for another command.

Upon initial start-up, the LNW-CP/M system is brought in from disk "A", and the CCP displays the message:

CP/M Ver. 2.2
Copyright (c) 1979, Digital Research
LNW Ver m.m
by R.M.Stiles

Where m.m is the LNW implementation version number. Following system sign-on, CP/M automatically logs in disk "A", and prompts the user with the symbol " A> " (indicating that CPM is currently addressing disk "A"), and waits for a command. The commands are implemented at two levels: Built-in commands and Transient commands.

GENERAL COMMAND STRUCTURE:

Built in commands are a part of the CCP program itself, while transient commands are loaded into the TPA from disk and executed. The built in commands are:

ERA - Erase specified files

DIR - List file names in the directory

REN - Rename the specified file

SAVE - Save memory contents in a file

TYPE - Type the contents of a file on the logged disk

Most of the commands reference a particular file or group of files.

FILE REFERENCES:

File reference identifies a particular file or group of files on a particular disk attached to CP/M. These file references are either "unambiguous" (ufn) or "ambiguous" (afn). An unambiguous file reference uniquely identifies a single file, while an ambiguous file reference is satisfied by a number of different files.

File reference consist of two parts: the primary filename and the filetype.The two names are separated by a ".", as shown below:

filename.typ

Where filename is the primary filename of eight characters or less, and typ is the filetype of no more than three characters. Although the filetype is optional, it usually is generic; that is, the filetype "ASM", for example, is used to denote that the file is an assembly language source file, while the primary filename distinguishes each particular source file. As mentioned above, the name:

filename

is also allowed and is equivalent to a filename consisting of three blanks. The characters used in specifying an unambiguous file reference cannot contain any of the special characters:

< > . , ; : = ? * [ ] _ % ( ) /

While all alphanumerics and remaining special characters are allowed.

An ambiguous file reference is used for directory search and pattern matching. The form of an ambiguous file reference is simular to an unambiguous reference, except the symbol "?" can be interspaced throughout the primary and secondary names. In various commands throughout CP/M, the "?" symbol matches any character of a file name in the "?" position. Thus, the ambiguous reference:

X?Z.C?M

is satisfied by the unambiguous filenames

XYZ.COM
X3Z.CAM

Note that the ambiguous reference

.

is equivalent to the ambiguous file reference

????????.???

while

filename.* and *.typ

are abbreviations for

filename.??? and ????????.typ

As an example,

A>DIR *.*

is interpreted by the CCP as a command to list the names of all disk files in the directory, while A>DIR X.Y searches only for a file by the name X.Y. Similarly, the command:

A>DIR X?Y.C?M

causes a search for all unambiguous file names on the disk that satisfy this ambiguous reference.

The following file names are valid unambiguous file references:

X X.Y XYZ XYZ.COM GAMMA GAMMA.I

As an added convenience, the user can generally specify the disk drive name along with the file name. In the case, the drive name is given as a letter A through D followed by a colon (:). The specified drive is "logged in" before the file operation occurs. Thus, the following are valid file names with disk name prefixes:

A:X.Y B:X.Y C:XYZ D:XYZ.COM

All alphanumeric lower case letters in file and drive names are translated to upper case when they are processed by the CCP.

SWITCHING DISKS:

The operator can switch the currently logged disk by typing the disk drive name (A through D) followed by a colon (:) when the CCP is waiting for console input. Thus, the following sequence of prompts and commands can occur after CP/M is loaded from disk A:

```
A>dir                          = list all files on A

A:sample  ASM   sample   PRN

A>b:                           = switch to disk B

B>dir *.asm                    = list all "ASM" files on B

B:Dump.ASM      files.ASM

B>A:                           = switch to disk A
```

BUILT IN COMMANDS:

The file and device reference forms described can now be used to fully specify the structure of the built in commands. The user should assume the following abbreviations in the descriptions below:

ufn = Unambiguous file references

afn = Ambiguous file references

Remember -- CCP always translates lower case characters to upper case characters internally.

ERA afn:

The ERA (erase) command removes files from the currently logged disk. The files that are erased are those that satisfy the ambiguous file reference "afn". The following examples illustrate the use of ERA:

ERA X.Y =	Erases file named X.Y
ERA X.* =	Erases all files with primary name X
ERA *.ASM	Erases all files with extension name ASM
ERA X?Y.C?M	Erases all files that satisfy the ambiguous reference X?Y.C?M
ERA *.*	Erases all files on disk (in this case the CCP prompts with the message 'ALL FILES (Y/N?)) This requires a Y before any files are removed.
ERA B:*.PRN	All files on drive B that satisfy the ambiguous reference ????????.PRN are erased independently of the current logged disk.

DIR afn:

The DIR (directory) command causes the names of all files that satisfy the ambiguous file name afn to be listed at the console device. As a special case, the command DIR lists the files on currently logged disk (the DIR is equivalent to DIR *.*). Examples of DIR commands are:

```
DIR X.y

DIR X?Y.C?M

DIR ??.Y
```

As with the CCP commands, the afn can be preceded by a drive name:

```
DIR B:

DIR B:X.Y

DIR C:*.A?M
```

If no files on the selected disk satisfy the DIR request, the message "no file" appears on the screen.

REN ufnl=ufn2:

The REN (rename) command allows the user to change the names of files on disk. The file satisfying ufn2 is changed to ufnl. The following are examples:

REN NEWNAME=OLDNAME	The file OLDNAME is changed to NEWNAME.
REN X.Y=Q.R	The file Q.R is changed to X.Y
REN XYZ.COM=XYZ.XXX	The file XYZ.XXX is changed to XYZ.COM

ufnl or ufn2 or both can be preceded by an optional drive name. If ufnl is preceded by a drive name, then ufn2 is assumed to exist on the same drive. Similarly, if ufn2 is preceded by a drive name, then ufnl is assumed to exist on that drive as well. The same drive must be specified in both cases if both ufnl and ufn2 are preceded by drive names. The following are examples:

REN A:X.ASM=Y.ASM	File Y.ASM is changed to X.ASM on drive A
REN B:ZAP.BAS=ZOT.BAS	File ZOT.BAS is changed to ZAP.BAS on drive B
REN B:X.ASM=B:X.BAK	File X.BAK is changed to X.ASM on drive B

If ufnl is already present, the REN command will respond with the error "file exists" and not confirm the change. If ufn2 does not exist, the message "no file" appears.

SAVE n ufn:

The save command places n pages (256-byte blocks) onto disk from the TPA and names this file ufn. TPA starts at 100h (hexidecimel) which is the second page of memory. The SAVE command must specify 2 pages of memory if the user's program occupies the area from 100H to 2FFH. The machine code file can be subsequently loaded and executed. Examples are:

SAVE 3 X.COM	Copies 100H through 3FFH to X.COM
SAVE 40 Q	Copies 100H through 28FFH to Q (note that 28 is the page count in 28FFH, and that 28H = 2 * 16 + 8= 40 decimal)
SAVE 4 S.Y	Copies 100H through 4FFH to S.Y

The SAVE command can also specify a disk drive in the ufn portion of the command, as shown:

SAVE 10 B:ZOT.COM	Copies 10 pages (100H through 0AFFH) to the file ZOT.COM on drive B

TYPE ufn:

The TYPE command displays the contents of the ASCII source file ufn on the currently logged drive.

```
TYPE X.Y

TYPE X.PLM

TYPE XYZ
```

are examples of the TYPE command.

The TYPE command expands tabs (ctl-1 characters), assuming tab positions are set at every eighth column. The ufn can also reference a drive name.

```
TYPE B:X.PRN
```

displays the file X.PRN from drive B.

USER n

The USER command allows maintenance of separate files in the save directory and take the form:

```
USER n
```

where n is an integer value in the range 0 to 15. On cold start, the operator is automatically "logged" into user area 0, which is compatible with standard CP/M 1 directories. The operator may use the USER command at any time to move to another logical area within the same directory. Drives that are logged-in while addressing one user number are automatically active when the operator moves to another. A user number is simply a prefix that accesses particular directory entries on the active drive.

The user number is maintained until changed by a subsequent USER command, or until a cold start when user 0 is again assumed.

Line Editing and Output Control:

The CCP allow certain line editing functions while typing command lines.

Ctl-C	System Re-boot when typed at the start of line.
Ctl-E	Physical end of line: carriage is returned, but line return is not sent until the carriage return key is depressed.
Ctl-H	Back-space on character position.
Ctl-J	Terminate current line input (line feed).
Ctl-M	Terminate current line input (carriage return).
Ctl-R	Retype current command line.
Ctl-U	Delete the entire line typed at the console.
Ctl-X	Same as Ctl-U.
Ctl-Z	End input from the console (used in Pip and Ed).

The control functions Ctl-P and Ctl-S affect the console output.

Ctl-P	Copy all subsequent console output to the list device. Output is sent to the list device and the console until the next Ctl-P is typed.
Ctl-S	Stop the console output temporarily. Program execution and output continue when the next character is typed at console.

The control functions Ctl-9 and Ctl-0 affect the console only.

Ctl-9	Clears the screen - has no affect upon program under execution
Ctl-0	Toggles from the underline cursor (_) to a block (■) cursor.

The CTL-KEY sequences are obtained by depressing the control key and letter (number) keys simultaneously. Further, the CCP command lines are generally up to 255 characters in length; they are not acted upon until the carriage return key is typed.

TRANSIENT COMMANDS:

Transient commands are loaded from the currently logged disk and executed in the TPA. The transient commands for execution under the CCP are below:

STAT	List the number of bytes of storage remaining on the currently logged disk, provide statistical information about particular files, and display or alter device assignment.
ASM	Load the CP/M assembler and assemble the specified program from disk.
LOAD	Load the file in Intel "HEX" machine code format and produce a file in machine executable form that can be loaded into the TPA (this loaded program becomes a new command under the CCP).
DDT6	Load the CP/M debugger into TPA and start execution (6 indicates patch to DDT to move the interrupt from RST 7 to RST 6).
PIP	Load the peripheral interchange program for subsequent disk file and peripheral transfer operations.
ED	Load and execute the CP/M text editor program.
SUBMIT	Submit a file of commands for batch processing.
MOVCPM	Regenerate the CP/M system.
DUMP	Dump the contents of a file in Hex.
LNW	Allows changes to operating system parameters.
FMT	Formats diskettes and installs operating system.
SET	Allows manually changing the disk parameters within the BIOS.

Transient commands are specified in the same manner as built in commands, and additional commands are easily defined by the user. The transient command can be preceded by a drive name that causes the transient to be loaded from the specified drive into the TPA for execution. The command:

```
B:STAT
```

causes CPM to temporarily "log in" drive B for the source of the STAT transient, and then return to the original logged disk for subsequent processing.

STAT

The STAT command provides general statistical information about file storage and device assignment. It is initiated by one of the following forms:

STAT

STAT "Command Line"

Special forms of the "command line" allow the current device assignment to be examined and altered. The various command lines that can be specified are shown, with an explanation on each to the right.

STAT — If the user types an empty command line, the STAT transient calculates the storage remaining on all active drives, and prints the message:

d:R/W,SPACE:nnnK

or

d:R/O,SPACE:nnnK

for each active drive d:, where R/W indicates the drive can be read or written, and R/O indicates the drive is read only (a drive becomes R/O by explicitly setting it to read only, as shown below, or by inadvertently changing diskettes without performing a warm boot). The space remaining on the diskette in drive d: is given in kilobytes by nnn.

STAT d: — If a drive name is given, then the drive is selected before the storage is computed. Thus, the command "STAT B:" could be issued while logged into drive A, resulting in the message:

BYTES remaining on B: nnnK

STAT afn — The command line can also specify a set of files to be scanned by STAT. The files that satisfy afn are listed in alphabetical order, with storage information for each file under the heading:

RECS BYTS EX D:FILENAME.TYP

rrrr bbbk ee d:filename.typ

Where rrrr is the number of 128 byte records allocated to the file (bbb=rrr*128/1024), ee is the number of 16K extents (ee=bbb/16), d is the drive name containing the file (A...D), filename.typ is the name of the file.

STAT d:afn — The drive name can be given ahead of the afn. The specified drive is first selected, and the form "STAT afn" is executed.

STAT d:=R/O — This form sets the drive given by d to read only, remaining in effect until the next warm or cold start takes place. When a disk is read only, the message:

BDOS ERR ON d: READ ONLY

will appear if an attempt to write to the R/O disk d: CPM waits until a key is depressed before performing an automatic warm boot (at which time d: becomes R/W).

Within the LNW-CPM system, the STAT command that allows control over physical to logical devices affects only the IOBYTE (location 0003H).

The device assignments are:

CON:	Permanently assigned to the crt device
RDR:	Not implemented
PUN:	Not implemented
LST:	Permanently assigned to system printer (serial or parallel, see LNW utility)

The command

STAT VAL:

produces a summary of the available status commands, resulting in the output:

```
Temp R/O Disk d: $R/O

Set Indicator:filename.typ  $R/O  $R/W  $SYS  $DIR

Disk Status: DSK: d:DSK
iobyte assign:

CON: = TTY:CRT:BAT:UC1
RDR: = TTY:PTR:UR1:UR2
PUN: = TTY:PTP:UP1:UP2
LST: = TTY:CRT:LPT:UL1
```

The command STAT DEV: produces a list of device assignments set by the iobyte upon initialization.

```
CON: = CRT
RDR: = TTY
PUN: = TTY
LST: = LPT
```

Although the command:

STAT Log Devl = Phy Devl, Log Dev2 = Phy Dev2

will change the listing and the iobyte, it will have no effect on device assignments within the LNW-CPM CBIOS (CON=CRT,LST=LPT). The command form:

STAT d:filename.typ $S

where "d:" is an optional drive name and "filename.typ" is an unambiguous or ambiguous filename, produces the sample output display formats:

```
  SIZE      RECS       BYTS     EXT ACC
    48        48         6K       1  R/O A:ED.COM
    55        55        12K       1  R/O A:PIP.COM
 65536       128        16K       2  R/W A:X.DAT
```

Where the $S parameter causes the size field to be displayed. The size field lists the virtual file size in records, while the "RECS" field sums the number of virtual records in each extent. For files constructed sequentially, the size and recs field are identical. The "BYTES" field lists the actual number of bytes allocated to the file. The minimum allocation unit is determined at configuration time; thus, the number of bytes corresponds to the record count plus the remaining unused space in the last allocated block for sequential files. Random access files are given data areas only when written, so the bytes field contains the only accurate allocation figure. In the case of random access, the size field gives the logical end-of-file record position and the recs field counts the logical records of each extent. (each of these extents, however, may contain unallocated "holes" even though they are added into the record count.) The "EXT" field counts the number of physical extents allocated to the file. The EXT count corresponds to the number of directory entries given to the file.

The "ACC" field gives the R/O or R/W file indicator that is changed using commands shown. Similarly, the parenthesis shown about the PIP.COM filename indicate that it has the "system" indicator set, so it will not be listed in DIR commands. The four command forms

STAT d:filename.typ $R/O

STAT d:filename.typ $R/W

```
STAT d:filename.typ $SYS

STAT d:filename.typ $DIR
```

set or reset the various permanent file indicators. The R/O indicator places the file in a read-only status until changed by a subsequent STAT command. The R/W indicator change the file to read/write status. The SYS indicator attaches the system indicator to the file, while the DIR removes it. The filename.typ may be ambiguous or unambiguous.

The command form:

```
STAT d: DSK:
```

lists the drive characteristics of the disk "d:". The drive characteristics are listed in the format:

```
   d:  Drive Characteristics
1328:  128 Byte Record Capacity
 166:  Kilobyte Drive Capacity
  64:  32 Byte Directory Entries
  64:  Checked Directory Entries
 256:  Records/Extent
  16:  Records/Block
  36:  Sectors/Track
   3:  Reserved Tracks
```

Where d: is the selected drive, followed by the total record capacity (1328 is a 40-track 5 1/4" SSDD drive), followed by the total capacity listed in kilobytes. The directory size is listed next, followed by checked entries. The number of checked entries is identical to the directory entry. This is used to detect changed media during CP/M operation. The number of records per extent determines the addressing capacity of each directory entry The number of records per block shows the basic allocation size. The two remaining entries show the number of physical sectors per track and the number of reserved tracks.

The command form:

```
STAT DSK:
```

produces a drive characteristic table for all currently active drives.

The final Stat command form is:

```
STAT USR:
```

which produces a list of the user numbers that have files on the currently addressed disk. The display format is:

```
Active User :0
Active files :0 1 3
```

Where the first line lists the currently addressed user number followed by a list of user numbers having active files or the current disk.

ASM ufn

The ASM command loads and executes the CP/M 8080 assembler. The ufn specifies a source file containing assembly language statements where the file type is assumed to be ASM and is not specified. The following ASM commands are valid:

```
ASM X        for        X.ASM

ASM GAMMA    for        GAMMA.ASM
```

The two-pass assembler is automatically executed. Assembly errors that occur during the second pass are printed at the console.
The assembler produces a file:

```
X.PRN
```

The PRN file contains a listing of the source program, along with the machine code generated for each statement and diagnostic error messages, if any, The file:

```
X.HEX
```

is also produced, which contains 8080 machine language in INTEL "HEX" format suitable for subsequent loading. Complete details of the CP/M Assembler are beyond the scope of this manual, and can be found in the CP/M operations manual.

LOAD ufn

The LOAD command reads the file ufn, which is assumed to contain "HEX" format machine code, and produces a memory image file that can subsequently be executed. The filename ufn is assumed to be of the form:

```
X.HEX

GAMMA.HEX
```

and only the filename X or GAMMA, in the samples, need be specified in the command. The LOAD command creates a file named:

```
X.COM

GAMMA.COM
```

that marks it as containing machine executable code.

Generally the CCP reads the filename, X or GAMMA, following the prompting character ">" and looks for a built in function name. If no function name is found, the CCP searches the system disk directory for a file by the name X.COM or GAMMA.COM. If found, the machine code is loaded into the TPA, and the program executes. The user need only LOAD a hex file once; it can be subsequently executed any number of times. The operation takes place on an alternative drive if the file name is prefixed by a drive name.

```
LOAD B:BETA
```

brings the LOAD program into TPA from the currently logged disk and operates upon drive B after execution begins.

DDT6

The DDT6 program is the CP/M dynamic debugging tool. The 6 denotes that the restart instruction internally used by DDT, has been changed to restart 6 for the LNW-CP/M system. Its use is otherwise identical to the standard CP/M DDT program. DDT6 is invoked by one of the following forms:

```
DDT6

DDT6 Filename.Typ
```

Where "filename" is the name of the program to be loaded and tested. In both cases, the DDT6 program is brought into main memory in place of the CCP; and resides directly below the BDOS portion of CP/M. The BDOS starting address, located at the jump location 5H, is altered to reflect the reduced TPA area size.

The second form of the DDT6 command performs the same actions as the first, except there is a subsequent load of the specified file. The action is identical to the sequence of command:

```
DDT6
 IFilename.TYP
 R
```

Where the I and R commands set up and read the specified program. (the I and R commands are described below.) Upon initialization, DDT6 prompts the user with the character "_" and waits for input commands from the console. The operator can type any of several single character commands, terminated by a carriage return to execute the command. Each line of input can be line - edited using the standard CP/M controls:

Backspace	Remove last character typed
CTL-U	Remove the entire line, ready for retyping

Any command can be up to 32 characters in length (an automatic carriage return is inserted as the 33rd character), where the first character determines the command type.

A	Enter 8080 assembly language mnemonics with operands.
D	Display memory in hexidecimal and ASCII.
F	Fill memory with constant data.
G	Begin execution with optional breakpoints.
I	Set up a standard input file control block.
L	List memory using 8080 assembler mnemonics.
M	Move a memory segment from source to destination.
R	Read program for subsequent testing.
S	Substitute memory values.
T	Trace program execution.
U	Untrace program monitoring.
X	Examine and optionally alter the CPU state.

The command character, in some cases, is followed by zero, one, two, or three hexidecimal values, which are separated by commas or single blank characters. All DDT6 numeric output is in hexidecimal form.

At any point in the debug run, the operator can stop execution of DDT6 by using either a CTL-C or G0 (jmp to location 0000H), and save the current memory image by using a SAVE command of the form:

```
SAVE n filename.typ
```

Where n is the number of pages (256 byte blocks) to be saved on disk. The number of blocks is determined by taking the high order byte of the address in the TPA and converting this number to decimal. For example, if the highest address in the TPA is 1234H, the number of pages is 12H or 18 in decimal. The operator could type a CTL-C during the debug run, returning to the CCP level, followed by:

```
SAVE 18 X.COM
```

The memory image is saved as X.COM and can be directly executed by typing the name X.

A complete description of the DDT6 commands can be found in the CP/M operations manual.

PIP

PIP is the Peripheral Interchange Program that implements the basic media conversion operations necessary to load, print, copy and combine disk files. The PIP program is initiated by typing one of the following forms:

```
PIP

PIP "command line"
```

In both cases PIP is loaded into the TPA and executed. In the first form, PIP reads command lines directly from the console, prompted with the "*" character, until an empty command line is typed (ie. a single carriage return). The second form is equivalent to the first, except that the single command line given with the PIP command is automatically executed, and PIP terminates immediately with no further prompting. The form of each command line is:

```
destination=source#1, source#2,...,source#n
```

Where "destination" is the file or peripheral device to receive the data and "source #1,...,source #n" is a series of one or more files on devices that are copied from left to right to the destination.

When multiple files are given in the command line (ie. n>1), the individual files are assumed to contain ASCII characters, with an assumed CP/M end of file character (CTL-Z) at the end of each file (see the O parameter to override this assumption). Lower case ASCII alphabetics are internally translated to upper case to be consistent with CP/M file and device name conventions. The total command line length cannot exceed 255 characters (CTL-E can be used to force a physical carriage return for lines that exceed the console width).

The destination and source elements are unambiguous references to CP/M source files with or without a preceding disk drive name. When the drive name is not included, the currently logged disk is assumed. The destination file can also appear as one or more of the source files, in which case the source file is not altered until the entire concatenation is complete. If it already exists, the destination file is removed if the command line is properly formed (it is not removed if an error condition arises). The following command lines are valid as input to PIP:

X=Y Copy to file X from file Y, where X and Y are unambiguous file names; Y remains unchanged.

X=Y,Z Concatenate files Y and Z and copy to file X, with Y and Z remaining unchanged.

X.ASM=Y.ASM,Z.ASM,FIN.ASM

Create the file X.ASM from the concatenation of the Y,Z, and FIN files.

NEW.ZOT=B:OLD.ZAP

Move a copy of OLD.ZAP from drive B to the currently logged disk and name it NEW.ZOT. OLD.ZAP will remain unchanged on drive B.

B:W.ZAP=B:X.ZAP,A:Y.ZAP,Z.ZAP

Concatenate file X.ZAP from drive B with Y.ZAP from drive A and Z.ZAP from the logged disk; create the file W.ZAP on drive B.

For convenience, PIP allows abbreviated commands for transferring files between disk drives. The abbreviated forms are:

PIP B:=afn Copies all files from the currently logged disk that satisfy the afn to the same files on drive B.

PIP A:=B:afn Same as first example except copies from drive B to drive A

PIP ufn=C: Equivalent to PIP d:ufn=C:ufn where d is the currently logged disk. The file, ufn, is copied from drive C to logged disk.

PIP B:ufn=C: Copies the file ufn from drive C to drive B.

The source and destination disks must be different in all of these cases. If an afn is specified, PIP lists each ufn that satisfies the afn as it is being copied. If a file exists by the same name as the destination file, it is removed on successful completion of the copy and replaced by the copied file. The following PIP commands give examples of valid disk to disk copy operations:

B:=*.COM Copy all files that have the secondary name "COM" from logged drive to drive B.

A:=B:ZAP.* Copy all files that have the primary name "ZAP" from drive B to drive A.

ZAP.ASM=B: Copy ZAP.ASM from drive B to logged drive

B:ZOT.COM=A: Copy ZOT.COM from drive A to drive B

B:=GAMMA.BAS Copy GAMMA.BAS from logged drive to drive B

B:=A:GAMMA.BAS Copy GAMMA.BAS from drive A to drive B

PIP allows reference to physical and logical devices that are attached to the LNW-CPM System (CON and LST), along with a number of specially named devices. The additional device names that can be used in PIP commands are:

NUL: Send 40 "NULLS" (ASCII 0's)

EOF: Send a CP/M end-of-file (ASCII CTL-Z) Sent automatically at the end of all ASCII data transfers through PIP.

PRN: Same as LST: except that tabs are expanded at every eighth character position, lines are numbered, and page ejects are inserted every 60 lines with an initial eject (same as using PIP options (T8NP)).

File and device names can be interspersed in the PIP commands. In each case, the specific device is read until end-of-file (CTL-Z for ASCII files, and end-of-data for non-ASCII disk files). Data from device or file are concatenated from left to right until the last data source has been read. The destination device or file is written using the data from the source files, and an end-of-file character (CTL-Z) is appended to the result for ASCII files. If the destination is a disk file, a temporary file is created ($$$ secondary name) that is changed to the actual name only on successful completion of the copy. Files with the extension "COM" are always assumed to be non-ASCII.

The copy operation can be aborted at any time by depressing any key on the keyboard (a return suffices). PIP will respond with the message "ABORTED" to indicate that the operation has not been completed. If any operation is aborted, or if an error occurs during processing, PIP removes any pending commands that were set up while using the SUBMIT command.

Valid PIP commands are:

PIP LST:=X.PRN Copy X.PRN to the printer and terminate the PIP program.

PIP Start PIP for a sequence of commands (PIP prompts with "*").

*CON:=X.ASM,Y.ASM,Z.ASM

Concatenate three ASM files and copy to the console.

*X.HEX=CON:,Y.HEX Create a file by reading the console (until a CTL-Z is typed), followed by data from Y.HEX.

(carriage return) Single carriage return stops PIP

The user can also specify one or more PIP parameters, enclosed in left and right brackets (f1 key ([) for left bracket and f2 key (]) for right bracket), separated by zero or more blanks. Each parameter affects the copy operation, and the enclosed list of parameters must immediately follow the affected file or device. Generally, each parameter can be followed by an optional decimal integer value (the S and Q parameters are exceptions). Valid PIP parameters are:

B Block mode transfer : data are buffered by PIP until an ASCII x-off character (CTL-S) is received form the source device.

Dn Delete characters that extend past column n in the transfer of data to the destination from the source. This parameter is generally used to truncate long lines that are sent to a narrow printer or console.

E Echo all transfer operations to the console as they are being performed.

F Filter form feeds from the file. All embedded form feeds are removed. The P parameter can be used simultaneously to insert new form feeds.

Gn Get file from user number n

H Hex data transfer : all data are checked for proper Intel hex file format. Non-essential characters between hex records are removed during the copy operation. The console will be prompted for corrective action in case errors occur.

I Ignore ": " records in the transfer of intel hex format file (the I parameter automatically sets the H parameter).

L Translate upper case alphabetics to lower case.

N Add line numbers to each line transferred to the destination, starting at one and incrementing by 1. Leading zeros are suppressed, and the number is followed by a colon. If N2 is specified, leading zeros are included and a tab is inserted following the number. The tab is expanded if T is set.

O Object file (non-ASCII) transfer : the normal CP/M end-of-file is ignored.

Pn Include page eject every n lines (with an initial page eject). If n=1 or is excluded altogether, page ejects occur every 60 lines. If the F parameter is used, form feed suppression takes place before the new page ejects are inserted.

Qs z Quit copying from the source device on file when the string s (terminated by CTL-Z) is encountered.

R Read system files

Ss z Start copying from the source when the string s (terminated by CTL-Z) is encountered. The S and Q parameters can be used to "abstract" a particular section of a file (such as a subroutine). The start and quit strings are always included in the copy operations.

If the user selects form (2) of the PIP command, the CCP translates strings following the S and Q parameters to upper case translation:

(1) PIP

(2) PIP "command line"

Tn Expand tabs (CTL-L characters) to every nth column during the transfer of characters to the destination to source.

U Translate lower case alphabetic to upper case during the copy operation.

V Verify that data have been copied correctly by rereading after the write operation (the destination must be a disk file).

W Write over R/O files without console interrogation.

Z Zero the parity bit on input for each ASCII character.

Valid PIP commands that specify parameters in the file transfer are:

```
PIP  X.ASM=B:[V]
```

copy object file X.COM from the drive B to the current drive and verify.

```
PIP LST:X.ASM[NT8u]
```

Copy X.ASM to the printer; number each line, expand tabs to every eighth column, and translate lower case to upper case.

```
PIP X.LIB=Y.ASM [sSUBRI:z qJP   L3 z]
```

Copy from the file Y.ASM into the file X.LIB. Start the copy when the string "SUBRI:" has been found, and quit copying after the string "Jp L3" is encountered.

```
PIP PRN:=X.ASM[p50]
```

Send X.ASM to the printer with line numbers, tabs expanded to every eighth column, and page ejects at every 50th line. The assumed parameter list for a PRN file is nt8p60, p50 overrides the default value.

Under normal operation, PIP will overwrite a file that is set to a permanent R/O status. If an attempt is made to overwrite an R/O file, the prompt:

DESTINATION FILE IS R/O, DELETE (Y/N)?

is displayed. If the operator responds with "Y" the file is overwritten. Otherwise, the response:

** NOT DELETED **

is displayed, the file transfer is skipped and PIP continues with the next operation in sequence. To avoid the prompt and response in the case of R/O file overwrite, the command line can include the W parameter:

```
PIP A:=B:*.*[W]
```

which copies all non system files to the A drive from the B drive and overwrites any R/O files. If the operation involves several concatenated files, the W parameter need only be included with the last file in the list.

Files with the system attribute can be included in PIP transfers if the R parameter is included; otherwise, system files are not recognized. The command line:

```
PIP A:=B:*.*[R]
```

copies all files, including system files.

NOTE: To copy files into another user area, PIP.COM must be located in that user area. Follow the procedure shown below to make a copy of PIP.COM in another user area.

```
USER 0                  Log in user 0

DDT6 PIP.COM            (note PIP size s)  Load PIP to memory

G 0                     Return to CCP
```

USER 3	Log in user 3
SAVE s PIP.COM	Makes file PIP.COM in User 3

Where "s" is the integral number of memory "pages" (256 byte segments) occupied by PIP. The number s can be determined when PIP.COM is loaded under DDT6, by referring to the hexidecimal value under the NEXT display. If, for example, the next available address is 1D00, then PIP.COM requires one less than the high byte (1D-1=1C). 1C Hex = 28 decimal thus the command would be:

SAVE 28 PIP.COM

Once PIP is copied in this manner, it can be copied to another disk belonging to the same user number through normal PIP transfers.

ED ufn

The ED program is the CP/M system context editor that allows creation and alteration of ASCII files in the CP/M environment.

Complete details of ED operation are given in the CP/M operations manual and is beyond the scope of this document.

ED allows the operator to create and operate upon source files that are organized as a sequence of ASCII characters. If the file ufn does not exist, ED creates and opens it for access. If the source file does exist (see the "A" command), the programmer "appends" data for editing. The appended can then be displayed, altered, and written from the work area back to the disk (see the W command). Particular points in the program can be automatically paged and located by context (see the N command), allowing easy access to particular portions of a large file. Given that the operator has typed:

ED X.ASM

the ED programs create an intermediate work file with the name:

X.$$$

to hold the edited data during the ED session. Upon completion of ED, the X.ASM file (original file) is renamed to X.BAK, and the edited work file is renamed to X.ASM. The operator can always return to the previous version of a file by removing the most recent version and renaming the previous version. If the current X.ASM file has been improperly edited, the sequence of commands below will reclaim the back-up file:

DIR X.*	Check to see that BAK file is available.
ERA X.ASM	Erase most recent version.
REN X.ASM = X.BAK	Rename the BAK file to ASM.

The ED program allows the user to edit the source on one disk and create the back-up file on another disk. The form of the ED command is:

ED ufn d:

Samples are:

ED A:X.ASM	Edit file X.ASM on drive A, with new file and back-up on drive A.
ED B:S.ASM A:	Edit the file X.ASM on drive B to the temporary file X.$$$ on drive A, after editing, change X.ASM on drive B to X.BAK and change X.$$$ on drive A to X.ASM.

SUBMIT ufn parm #1....parm #n

The SUBMIT command allows CP/M commands to be batched for automatic processing. The ufn given in the SUBMIT command must be the file name of a file that exists on the currently logged disk, with an assumed file type of "SUB." The SUB file contains CP/M prototype commands with possible parameter substitution. The actual parameters parm #1...parm #n are substituted into the prototype commands, and, if no errors occur, the file of substituted commands are processed sequentially by CP/M.

The prototype command file is created using an editor, such as ED in other word processing programs, with interspersed "$" parameters of the form:

$1 $2 $3...$n

corresponding to the number of actual parameters that will be included when the file is submitted for execution. When the SUBMIT transient is executed, the actual parameters parm #1 .... parm #n are paired with the formal parameters $1...$n in the prototype commands. If the numbers of formal and actual parameters do not correspond, the submit function is aborted with an error message at the console. The submit function creates a file of substituted commands with the name:

$$$.SUB

on the logged disk. When the system reboots (at the termination of the submit), this command file is read by the CCP as a source of input rather than the console. If the submit function is performed on any disk other than drive A, the commands are not processed until the disk is inserted into drive A and the system reboots. The user can abort processing at any time by typing a reboot (key) when the command is read and echoed. In this case the $$$.SUB file is removed and the subsequent commands come from

the console. Command processing is also aborted if the CCP detects an error in any of the commands. Programs that execute under CP/M can abort processing of command files when error conditions occur by erasing any $$$.SUB file.

The last command in a SUB file can initiate another SUB file, allowing chained batch commands.

Suppose the file ASMBL.SUB exists on disk and contains the prototype commands:

```
ASM $1

DIR $1.*

ERA *.BAK

PIP $2: = $1.PRN

ERA $1.PRN
```

and the command:

```
SUBMIT ASMBL X PRN
```

is issued by the operator. The submit program reads the ASMBL.SUB file, substituting "X" for all occurrences of $1 and "PRN" for all occurrences of $2. This results in a $$$.SUB file containing the commands:

```
ASM X

DIR X.*

ERA *.BAK

PIP PRN: = X.PRN

ERA X.PRN
```

which are executed in sequence by the CCP.

The SUBMIT function can access a sub file or an alternate drive by preceding the file name by a drive name. Submitted files are only acted upon when they appear on drive A. Thus it is possible to create a submitted file on drive B that is executed at a later time when inserted in drive A.

X.SUB

Is an additional utility program that extends the power of the SUBMIT facility to include line input to program as well as the CCP. The X.SUB command is included as the first line of the submit file. When it is executed, X.SUB self relocates directly below the CCP. All subsequent submit command lines are processed

by X.SUB so that programs that read buffered console input (BDOS function 10) receive their input directly from the submit file. For example, the file SAVER.SUB can contain the submit lines:

```
X.SUB

DDT6
  I$1.COM
  R
  G0

SAVE 1 $2.COM
```

with a subsequent submit command:

```
SUBMIT SAVER Myfile Y
```

That substitutes Myfile for $1 and Y for $2 in the command stream. The X.SUB program loads, followed by DDT6. DDT6 is then sent the command lines:

```
I MYFILE.COM
R
G0
```

which returns to the CCP. The final command:

```
SAVE 1 Y.COM
```

is processed by the CCP.

The X.SUB program remains in memory and prints the message:

```
(X.SUB ACTIVE)
```

on each warm start operation to indicate its presence. Subsequent submit command streams do not require the X.SUB, unless an intervening cold start has occurred.

NOTE: X.SUB must be loaded after the optional CP/M despool utility, if both are to run simultaneously.

MOVCPM

The MOVCPM program is not normally used with the LNW-CP/M system. It is included with the distributed diskette for special applications by the advanced programmer.

The use of MOVCPM follows normal conventions described in the CP/M manual with the exception of automatically configuring for maximum size. A size of 61K or less has to be passed as a parameter.

DUMP ufn

The DUMP program types the contents of the disk file (ufn) at the console in hexidecimal form. The file contents are listed sixteen bytes at a time, with the absolute byte address listed to the left of each line in hexidecimal. Long typeouts can be aborted by pushing the rubout (break) key during printout. The file DUMP.ASM is included on the distribution diskette as an example of a program written for the CP/M environment.

LNW

The LNW program allows the CBIOS to be changed for different drive configurations, serial, parallel, or custom printer driver selection, and 16 or 24 lines per screen display. The program makes immediate changes to CBIOS in memory. On exiting the program the user is given the choice of making all changes, except screen size, a permanent change by writing to the CBIOS on the system disk presumed to be in drive A.

The distribution diskette is set for 4 drives active, slowest step rate (3), parallel printer, and the 80x16 screen.

LNW

will display the following:

```
          Disk Parameter Change Facility by R.M. Stiles

4 Disks in system                                          Screen Size
 *Drive A Parameters = 5" 40 trk S/Side D/Den :step=3
 *Drive B Parameters = 5" 40 trk S/Side D/Den :step=3     <L> 80X24
 *Drive C Parameters = 5" 40 trk S/Side D/Den :step=3     <M> 80X16
 *Drive D Parameters = 5" 40 trk S/Side D/Den :step=3
  <1> Change number of drives in system
                                                           Printer

 Enter drive to change or menu item                      *<P>arallel (Nor)
                                                          <S>erial
                                                          <U>Custom
```

An asterisk indicates the parameters that are active. See appendix (E) for a sample session using the LNW utility. Selecting L or M from the main menu will change the screen size between 80x16 to 80x24. This change cannot be made permanent from the LNW utility, but will remain in memory until a cold restart is done, or changed by the LNW utility. Appendix (B) shows how to change the screen size permanently.

The program has two built in printer drivers, Serial and Parallel. When either of these are selected from the main menu, a second menu will give the choice of

Normal line feed	driver will send what is received.
Automatic line feed	driver will add a line feed (0AH) after each carriage return (0DH).
No line feed	driver will remove line feed after a carriage return.

The <U>custom selection is for a user installed custom driver. Appendix (D) gives a listing of each of the drivers and instructions for installing the custom driver.

FMT

The FMT program formats disks for the LNW-CP/M system. It will generate the system tracks configured for the target diskette, from the system diskette presumed to be in drive A. If the target diskette is drive A, then the user will be prompted for the correct disk to be inserted.

The FMT command has four forms, they are:

```
FMT

FMT SYS

FMT GET

FMT PUT
```

The command:

```
FMT
```

with no command tail, will format the entire target diskette and generate the system on the system tracks (if the target diskette is a system disk). The main menu:

```
LNW-CP/M Format Utility by R.M. Stiles

Which Drive (A,B,C,D)

Select one. (_)
```

will display the formats available to LNW-CP/M. Select the format for the target diskette. The display will ask for the target drive name. Select the drive to be formatted. The display will

now list the formats available. Select one. The display will now show the selection that was chosen, and ask for a Y/N response. A response of Y will start the format, and an N will return to program start. The formatting is done in three steps, write, read, and verify. These steps along with the track information are displayed as they are performed. The track is written on the write cycle, each sector is then read, during read cycle to verify correct track and sector identification. The verify cycle then checks the data (E5) that was written. If an error occurs an error message will be displayed and the last cycle shown will be where the error occurrred.

NOTE: At the response question of Y/N, responding with a zero (0) will cause the complete format without reading or verifying. This is much faster, but will not check the diskette for errors. It is not recommended unless you are very sure of your diskette.

At the completion of the format the screen will indicate:

READING SYSTEM

At this time the system from drive A is being read into memory and configured for the target diskette. The message:

WRITING SYSTEM

will then show as the system is being written to the target diskette. If no errors have occurred then the completion message:

COMPLETE

will be displayed, asking if you want to leave the program or format another diskette. If an error has occurred you will have the choice of exiting the program or returning to the start. At the successful completion, the target diskette is ready for use. It will have the system on the system tracks, but no files. Use the PIP command to transfer files.

The command:

FMT SYS

will format just the system tracks and generate the system. This form of the FMT command will not interfere with any files that may be on the diskette. It can be use to repair the system tracks and to apply future updates without having to transfer files.

The last two commands are used together. The command:

FMT GET

will read the system from the system diskette, assumed to be in drive A, and install it configured for the target system into

memory starting at loc 7000H. A memory map of the system at this location is shown in appendix (I). Modifications and changes can then be made using the DDT6 program.

WARNING: The byte at 6FFFH must not be altered. This byte passes information between the FMT GET and FMT PUT the target system.

The command:

FMT PUT

will write the system located at 7000H onto the system tracks of the target diskette. No formatting will be done. This form of the command is used in conjunction with the FMT GET form. The appendices show several examples using these two forms with DDT6 to make system changes.

SET

The SET utility allows the experienced user to manually change the drive parameter, and skew tables in the CBIOS. This gives the ability to be able to read and copy many different systems diskettes. The system can only read foreign diskettes with a maximum of 512 bytes per sector. Appendix (F) will show how to set the system to read larger sectors.

The command:

SET

will display:

Disk Parameter Change Facility by R.M. Stiles

Enter Drive to Change (A, B, C, or D) or X to exit

The screen will display the drive parameters presently set in the CBIOS for the selected drive. These parameters can now be changed. The CBIOS tables will not reflect these changes until the final Y/N response question. After completion of the parameter changes, a Y/N response is asked before preceding. A Y will advance to the skew tables. The table shown is that presently set in CBIOS for that drive. At the completion of this table, a Y/N response is again asked, a Y response here will set the changes within the CBIOS and return to the start of program. If the same drive is again selected the tables will show the changes just made.

The changes made within the set program are changed temporarily and will remain in effect until the next cold start. To make a permanent change, the LNW program must be used in conjunction with SET. Make the changes with SET. Enter the LNW program (it will recognize this change as a non-standard format). Toggle the printer selection (this sets the change flag within

the program). Exit the program by saving to disk. This disk will now have the new drive parameters set permanently.

BDOS Error Messages

There are three error situations that the BDOS intercepts during file processing. When one of these conditions is detected, the BDOS prints the message:

BDOS ERR ON d: error

where d is the drive name and "error" is one of the three error messages:

BAD SECTOR

SELECT

READ ONLY

The "BAD SECTOR" message indicates that the disk controller electronics has detected an error condition in reading or writing the diskette. Recovery from this condition is accomplished by typing a ctl-C to reboot (the safest course), or a return, which ignores the bad sector in the file operation.

The "SELECT" error occurs when there is an attempt to address a drive beyond the range supported by the LNW CP/M system. In this case, the value of d in the error message gives the selected drive. The system reboots following any input from the console.

The "READ ONLY" message occurs when there is an attempt to write to a diskette or file that has been designated as read only in a STAT command or has been set to read only by the BDOS. The operator should reboot CP/M by using the warm start procedure (ctl-C) or by performing a cold start whenever the diskettes are changed. If a diskette is to read but not written, BDOS allows the diskette to be changed without the warm or cold start, but internally marks the drive as read only. The status of the drive is subsequently changed to read/write if a warm or cold start occurs. On issuing the message, CP/M waits for input from the console. An automatic warm start takes place following any input.

TECHNICAL DESCRIPTION

The keyboard and video drivers are located in the alternate 32K of memory. Each call to BIOS for console input or output will bank switch these drivers in. The 80X16 screen format addresses the first 64 characters of the line through the character generator, The last 16 are addressed through the video character RAM tables located in the alternate memory bank. The 80X24 screen format uses the video RAM tables for all character generation. This method of addressing is the reason for the differences in speed between the two formats. The user can address the screen

area 64X16 directly by writing to locations FC00H to FF00H. A complete description of the LNW screen addressing methods can be found in the LNW reference manuals.

Caps Lock

The ctl-f2 keys toggle the caps-lock function. This function translates lower case to upper case on the alphabetic keys in the unshifted mode.

Cursor

The cursor is blinked each time the call to CBIOS const, or conin is made. If the cursor is blinking then the keyboard is being queried.

The cursor can be changed from the underline character to a block character at any time by the keys ctl-0. The ctl-0 will toggle between the two cursors.

VIDEO DRIVER

The video driver closely emulates a Lear Siegler ADM-3A terminal. This allows the user to install some complex software that asks for terminal type.

The cursor is controlled by a string of four bytes in the form

1B	escape hex 1BH
3D	equal hex 3DH
20+	20H + Line number
20+	20H + Column number

for example, if line 15, character position (column) 73 were to be addressed then the string would be

15 dec = 0F hex + 20 hex = 2F hex
73 dec = 49 hex + 20 hex = 69 hex

or

1B 3D 2F 69

Two functions that can be used by some programs that speed up operations are; Insert line and delete line. These functions can be accessed by the escape code sequence

1B 3D 00 08	=	delete the line that the cursor is on and move remaining lines up one.
1B 3D 00 09	=	insert line at the cursor position and move lines down one.

The following single byte commands will cause the specific

functions listed

0EH	=	turns cursor on.
0FH	=	turns cursor off.
14H	=	toggles the underline / block cursor.
1AH	=	clears the screen and homes cursor.
1EH	=	clears the line the cursor is on.
1FH	=	clears from the cursor to the end of the screen.

The large screen (80X24) will allow hi lighting. If bit 7 of the character to be displayed is set then it will be displayed in reverse video. This is ignored with the 80X16 screen.

The LNW has two speeds of operation, 1.77 mhz and 4 mhz. The LNW CP/M system with 5 " drives will operate at either speed. However, with 8" drives you MUST use the high speed (4 mhz).

INTERRUPTS

The LNW-CPM distribution diskette is set for interrupts disabled. The user can enable interrupts by changing the interrupt flag (byte EF49H) to zero (0). This byte is used in conjunction with the EI and DI instructions (set byte to 0 with EI and to non 0 with DI).

The system boots in interrupt mode 1. The user is responsible for the routine at location 38H if using interrupts.

A warm boot or cold start will return the interrupts to the disabled state.

WARNING: Although the LNW-CPM system will operate with interrupts enabled, some software will not. The user should beware running in the enabled state except with tested software.

APPENDIX A

Distribution diskette files:

MOVCPM.COM
STAT.COM
LNW.COM
S80X24.COM
S80X16.COM
PIP.COM
DDT6.COM
SUBMIT.COM
XSUB.COM
FMT.COM
SET COM
ED.COM
ASM.COM
LOAD.COM
DUMP.COM
CPMDVR.016
CPMDVR.024
DUMP.ASM

Permanent screen size change.

The following programs have to be on the disk in drive A:

```
CPMDVR.O16      for 80X16 screen format
CPMDVR.O24      for 80X24 screen format
FMT.COM
DDT6.COM
```

Follow the sequence below.

1. fmt get
   select the target system

2. ddt6
   icpmdvr.o24
   r7000
   g0

3. fmt put
   select the target drive

The diskette now has the 80X24 screen format permanently installed.

NOTE- The lnw program can still be used to switch between the two formats.

APPENDIX C

Console keyboard assignments.

The keyboard and video drivers are located in the alternate bank of memory, and can be easily altered using the FMT GET and FMT PUT commands. The following will list the key assignment locations, and their present assignments. A sample run-through will give an example of changing the arrow keys for the WORD STAR word processing program (Copyright MicroPro International).

KEY	UNSHFT LOC	ASN	SHFT LOC	ASN	CTL LOC	ASN
(f1)	733BH	5BH	733CH	7BH	733Dh	7CH
(f2)	733EH	5DH	733FH	7DH	7340H	FFH
(=)	7341H	2DH	7342H	5FH	7343H	1FH
(CR)	7344H	0DH	734CH	0DH	7354H	0DH
(clr)	7345H	15H	734DH	15H	7355H	15H
(brk)	7346H	7FH	734EH	7FH	7356H	1BH
(↑)	7347H	0AH	734FH	0AH	7357H	5BH
(↓)	7348H	0CH	7350H	0CH	7358H	5CH
(←)	7349H	08H	7351H	08H	7359H	5DH
(→)	734AH	09H	7352H	09H	735AH	5EH
(spc)	734BH	20H	7353H	20H	735BH	20H

The following key's ctl function can be altered.

KEY	CTL LOC	ASN	KEY	CTL LOC	ASN
(0)	735CH	9CH	(8)	7364H	5BH
(1)	735DH	7AH	(9)	7365H	9AH
(2)	735EH	7EH	(:)	7366H	1AH
(3)	735FH	1DH	(;)	7367H	1BH
(4)	7360H	1EH	(,)	7368H	1CH
(5)	7361H	1FH	(=)	7369H	1DH
(6)	7362H	7BH	(.)	736AH	1EH
(7)	7363H	7DH	(/)	736BH	5FH

The following example will change the arrow keys to the same function as the cursor movement keys in the Word Star word processing program.

1. fmt get
   select target system

2. ddt6
   s734f
   734F 0A 5
   7350 0C 18
   7351 08 13
   7352 09 4
   7353 20 .

   s7357
   7357 5B 12
   7358 5C 3
   7359 5D 1
   735A 5E 6
   735B 20 .

   g0

3. fmt put
   select target drive

The arrow keys will now operate as follows:

(shift ←)	=	ctl-s	(shift →)	=	ctl-d
(shift ↑)	=	ctl-e	(shift ↓)	=	ctl-x
(ctl ←)	=	ctl-a	(ctl →)	=	ctl-f
(ctl ↑)	=	ctl-r	(ctl ↓)	=	ctl-c

APPENDIX D

Printer driver routines

The location of the printer driver can be obtained from the cbios jump table at location EE0FH.

Serial driver:

```
   18 00  list:     jr   $+2            ;will be jr listll after
                                        ;after UART init.
   D3 E8            out  (pporta),a     ;out with anything
   DB E9            in   a,(pportb)     ;get LNW sense switchs
   E6 F8            and  0f8h           ;lop off lower 3 bits
   F6 04            or   4              ;resets rts, dtr
   D3 EA            out  (pportc),a     ;send to UART
   3E 0F            ld   a,0FH          ;dis to listll
32 xx xx            ld   (list+1),a     ;make jr change at list
   DB E8  listll:   in   a,(pporta)     ;chk handshaking
   CB 77            bit  6,a            ;is it ok
   20 FA            jr   nz,listll      ;loop back if not
   DB EA            in   a,(pportc)     ;chk next bit
   CB 77            bit  6,a            ;is it ok
   28 F4            jr   z,listll       ;go back if not
      79            ld   a,c            ;get char
-------------------------------------------------------------
AUTOLF:
   D3 EB            out  (pportd),a     ;send it out
   FE 0D            cp   0dh            ;was it a CR
      C0            ret  nz             ;rtn if not
   0E 0A            ld   c,0ah          ;get LF
   18 EA            jr   listll         ;go send it

NORLF:
   D3 EB            out  (pportd),a     ;send it out
      C9            ret                 ;return

NOLF:
   FE 0A            cp   0ah            ;is it LF
   20 07            jr   nz,nolf1       ;go if not
3A 44 EF            ld   a,(prtbyt)     ;get last byte sent
   FE 0D            cp   0dh            ;was it a CR
   28 03            jr   z,nolf2        ;go if it was
      79            ld   a,c            ;get byte to send back
   D3 EB  nolf1:    out  (pportd),a     ;send it
32 44 EF  nolf2:    ld   (prtbyt),a     ;save the byte
      C9            ret
```

Parallel driver:

```
3A E8 F7  list:     ld   a,(prtsta)     ;check for printer busy
   E6 F0            and  0F0h           ;lop lower 4 bits
   FE 30            cp   30h            ;is it ok
   20 F7            jr   nz,list        ;go back if not
      79            ld   a,c            ;get char
-------------------------------------------------------------
AUTOLF:
32 E8 F7            ld   (prtsta),a     ;send it out
   FE 0D            cp   0dh            ;was it CR
      C0            ret  nz             ;rtn if not
   0E 0A            ld   c,0ah          ;get LF
   18 EC            jr   list           ;go send it

NORLF:
32 E8 F7            ld   (prtsta),a     ;send it out
      C9            ret                 ;return

NOLF:
   FE 0A            cp   0ah            ;is it a lf
   28 08            jr   nz,nolf1       ;go if not
3A 44 EF            ld   a,(prtbyt)     ;get last byte sent
   FE 0D            cp   0dh            ;was it a CR
   18 04            jr   z,nolf2        ;go if so
      79            ld   a,c            ;get byte to send back
32 E8 F7  nolf1:    ld   (prtsta),a     ;send it out
32 44 EF  nolf2:    ld   (prtbyt),a     ;save the byte
      C9            ret
```

The Custom driver is all zero's on the distribution diskette. The user can install a driver in this location by following the procedure listed below.

1. The custom driver can be no longer then 47 bytes.

2. ddt6 lnw.com
   a. Load the custom driver, starting at location 104h.
   b. This can be done with the S function of ddt6, or if it has been loaded at another location then it can be moved with the M function. Warning: Make sure the driver is no longer then 47 bytes.
   c. g0 ;return to command level

3. save 15 lnw.com

4. The driver can now be loaded by using the LNW program.

APPENDIX E

Example using the LNW utility program.

The example will assume the following hardware system:

2 - 5" 40 trk S/Sided drives, step rate 20 ms
1 - 8" 77 trk D/Sided drive, step rate 3 ms
a Serial printer

Enter LNW (all instructions will show user input as < >.)

Appropriate menus will be assumed.

1- <A> =to set step rate drive A (main menu)
<C> =select menu item to change step rate (menu A)
<2> =20 ms for 5" drives (step menu)

2- <B> =to set step rate drive B (main menu)
<C> =select change step rate (menu A)
<2> =20 ms (step menu)
<B> =to set parameters (main menu)
<1> =5" 40 trk S/Side D/Den drive (menu A)

3- <C> =to set step rate drive C (main menu)
<C> =select change step rate (menu A)
<0> =3 ms for 8" drive (step menu)

4- <1> =change number of drives (main menu)
<3> =number of drives (# drv menu)

5- <S> =serial printer (main menu)

Check over your changes. If satisfied then exit the LNW program without making permanent changes. This will allow the user to check the system. If a fatal error has occurred, a cold start will remove the changes just made.

6- <X> =exit system (main menu)
<N> =not to be permanent

If satisfied with the changes, then re-enter the LNW program and make the changes permanent (if desired).

1- LNW

Toggle the printer to set change flag within system.

2- <P> <S>

3- <X> =exit system
<Y> =make changes permanent

The screen size cannot be made permanent with the LNW.

APPENDIX F

Disk I/O buffer size.

The LNW-CPM host buffer allows for a maximum of 512 bytes per sector. A larger foreign system can be read by modifying the location of the host buffer. This is done by moving the buffer down to a lower memory location. Care has to be taken, as this memory is not protected, and other programs may overwrite it. The buffer is presently located at FA00H.

Example: Set up for 1024 byte per sector.

Enter DDT6

Using ddt6, search the area of F242H for the bytes

11 00 FA

bottom of CCP = D800H therefore D800H - 1024 = D400H so change to

11 00 D4

search the area of F2F5H for the bytes

01 00 FA

change to

01 00 D4

When completed with operation using the larger buffer, change back to original for normal operation. A cold start will accomplish this.

APPENDIX G

CBIOS Jump Vectors

The CBIOS jump vectors for the LNW CP/M system are listed below:

```
EE00H     jp   boot      ;cold start entry
EE03H     jp   wboot     ;warm boot entry
EE06H     jp   const     ;console status check
EE09H     jp   conin     ;console char input
EE0CH     jp   conout    ;console char output
EE0FH     jp   list      ;printer output
EE12H     jp   punch     ;not used has a rtn
EE15H     jp   reader    ;not used rtns 1AH
EE18H     jp   home      ;head to trk 0
EE1BH     jp   seldsk    ;select disk
EE1EH     jp   settrk    ;set track #
EE21H     jp   setsec    ;set sector #
EE24H     jp   setdma    ;set dma address
EE27H     jp   read      ;read sector
EE2AH     jp   write     ;write sector
EE2DH     jp   listst    ;not used rtn 0
EE30H     jp   sctran    ;sector translate
```

These locations are fixed locations within the CBIOS and can always be used to determine routine locations.

APPENDIX H

INSTALLING WORDSTAR

This example of installing Wordstar is not the only or necessarily the optimum. It is given as an example of using the install program with the LNW-CP/M system.

1. Terminal ADM-3A (A)
2. Printer Your printer probably (A) or (C)
3. Comm Protocol None (N)
4. Driver CP/M List device (L)
5. Are mods complete? no

```
HITE:           =10     ;for 16 lines
                =18     ;for 24 lines

WID:            =50     ;for 80 column

ERAEOL:         =01     ;erase to end of line
ERAEOL:+1       =1E

LINDEL:         =04     ;delete line
LINDEL:+1       =1B
LINDEL:+2       =3D
LINDEL:+3       =00
LINDEL:+4       =08

LININS:         =04     ;insert line
LININS:+1       =1B
LININS:+2       =3D
LININS:+3       =00
LININS:+4       =09

TRMINI:         =01     ;clear  screen at beginning of
TRMINI:+1       =1A     ;Word Star session

TRMUNI:         =01     ;clear screen when exiting WS
TRMUNI:+1       =1A

HIBIV:          =FF     ;for 80X24 screen Hilite
HIBIV:          =00     ;for 80X16 screen NO Hilite
```

This routine will speed up the initialization of Word Star.

```
UCRPOS:         =C3         ;note UCRPOS address for later
UCRPOS:+1       =E0         ;use. Present version is 0264H
UCRPOS:+2       =02
```

The address at UCRPOS:+1 is the address of MORPAT:. In this version it is 02E0h. If yours is different then change UCRPOS+1, +2 accordingly.

```
MORPAT:         =3E
MORPAT:+1       =02
MORPAT:+2       =3D
MORPAT:+3       =32
MORPAT:+4       =E1*        ;address is MORPAT:+1
MORPAT:+5       =02*        ;check your version
MORPAT:+6       =C0
MORPAT:+7       =32
MORPAT:+8       =64*        ;address is UCRPOS:
MORPAT:+9       =02*
MORPAT:+10      =32
MORPAT:+11      =65*        ;address is UCRPOS:+1
MORPAT:+12      =02*
MORPAT:+13      =3E
MORPAT:+14      =C9
MORPAT:+15      =32
MORPAT:+16      =66*        ;address is UCRPOS:+2
MORPAT:+17      =02*
MORPAT:+18      =C9
```

The listing of the above routine is

```
MORPAT:   ld      a,2             ;we want to ignore
          dec     a               ;the first two cursor
          ld      (MORPAT+1),a    ;positioning calls
          ret     nz
          ld      (UCRPOS),a      ;now restore UCRPOS
          ld      (UCRPOS+1),a    ;to org 00 00 C9.
          ld      (UCRPOS+2),a
          ret
```

APPENDIX I

Memory Map of the LNW-CPM System after using the FMT GET command.

The FMT-GET will relocate the system to location 7000H.

6FFFH	Byte passes information between the FMT GET and the FMT PUT programs.
7000H	Boot loader
7100H	Sign-On Message
7200H	Driver information (Do not alter)
7300H	Keyboard and video drivers
7E00H	LNW-CPM System

7E00H is the base of the CCP. Its normal location is D800H.

```
*********************************************
*         FUNCTION 37:  RESET DRIVE         *
*********************************************
*       Entry Parameters:                   *
*           Register   C:  25H              *
*           Register  DE:  Drive Vector     *
*                                           *
*       Returned Value  :                   *
*           Register   A:  ØØH              *
*********************************************
```

The RESET DRIVE function allows resetting of specified drive(s). The passed parameter is a 16 bit vector of drives to be reset, the least significant bit is drive A.

In order to maintain compatability with MP/M, CP/M returns a zero value.

```
*********************************************
* FUNCTION 40: WRITE RANDOM WITH ZERO FILL  *
*********************************************
*       Entry Parameters:                   *
*           Register   C:  28H              *
*           Register  DE:  FCB Address      *
*       Returned Value  :                   *
*           Register   A:  Return Code      *
*********************************************
```

The WRITE RANDOM WITH ZERO FILL operation is similar to FUNCTION 34: with the exception that a previously unallocated block is filled with zeros before the data is written.